

Research Highlights (Required)

To create your highlights, please type the highlights against each `\item` command.

It should be short collection of bullet points that convey the core findings of the article. It should include 3 to 5 bullet points (maximum 85 characters, including spaces, per bullet point.)

- Investigated the use of Autoencoders to learn generic features from wearable devices.
- Proposed channel-wise ensemble approaches for Autoencoders to identify unseen falls.
- Developed threshold tightening approaches on reconstruction error of Autoencoders.
- Compared on two activity recognition datasets against standard one-class classifiers and demonstrated better performance.
-



Detecting Unseen Falls from Wearable Devices using Channel-wise Ensemble of Autoencoders

Shehroz S. Khan^{a,**}, Babak Taati^b

^aToronto Rehabilitation Institute, 550 University Ave, Toronto, ON, M5G 2A2, Canada

^bToronto Rehabilitation Institute, 550 University Ave, Toronto, ON, M5G 2A2, Canada
Department of Computer Science, University of Toronto

ABSTRACT

A fall is an abnormal activity that occurs rarely, so it is hard to collect real data for falls. It is, therefore, difficult to use supervised learning methods to automatically detect falls. Another challenge in using machine learning methods to automatically detect falls is the choice of features. In this paper, we propose to use an ensemble of autoencoders to extract features from different channels of wearable sensor data trained only on normal activities. We show that choosing a threshold as maximum of the reconstruction error on the training normal data is not the right way to identify unseen falls. We propose two methods for automatic tightening of reconstruction error from only the normal activities for better identification of unseen falls. We present our results on two activity recognition datasets and show the efficacy of our proposed method against traditional autoencoder models and two standard one-class classification methods.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

Falls are a major cause of both fatal and non-fatal injury and a hindrance in living independently. Each year an estimated 424,000 individuals die from falls globally and 37.3 million falls require medical attention (Organization, 2016). Experiencing a fall may lead to a fear of falling (Igual et al., 2013), which in turn can result in lack of mobility, less productivity and reduced quality of life. There exist several commercial wearable devices to detect falls (Pannurat et al., 2014); most of them use accelerometers to capture motion information. They normally come with an alarm button to manually contact a caregiver if the fall is not detected by the device. However, most of the devices for detecting falls produce many false alarms (El-Bendary et al., 2013). Automatic detection of falls is long sought; hence, machine learning techniques are needed to automatically detect falls based on sensed data. However, a fall is a rare event that does not happen frequently; therefore, during the training phase, there may be very few or no fall samples. Standard supervised classification techniques may not be suitable in

this type of skewed data scenario. Another issue regarding the use of machine learning methods in fall detection is the choice of features. Traditional activity recognition and fall detection methods extract a variety of domain specific features from raw sensor readings to build classification models (Ravi et al., 2005; Khan, 2016). It is very difficult to ascertain the number or types of features, specially in the absence of fall specific training data to build generalizable models.

To handle the problems of lack of training data from real falls and the difficulty in engineering appropriate features, we explore the use of Autoencoders (AE) that are trained only on normal activities. AEs can learn generic features from the raw sensor readings in order to identify unseen falls during testing. We present two ensemble approaches of AE that train on raw data of the normal activities from different channels of accelerometer and gyroscope separately and the results of each AE is combined to arrive upon a final decision. Typically, while using AE, the maximum of reconstruction error on the training set is considered as the threshold to identify an activity as abnormal. However, we experimentally show that such threshold may not be appropriate for detecting falls. We present two threshold tightening techniques to remove few outliers from the normal data. Then, either a new threshold is derived using inter-

^{**}Corresponding author. Tel.: +1 416-597-3422; Fax: +1 416-597-6201;
e-mail: shehroz.khan@uhn.ca (Shehroz S. Khan)

quartile range or by training a new AE on the training data with outliers removed. We show result on two activity recognition datasets that contain different normal activities along with falls from wearable sensors. The main contributions of the paper are 1) confirming the suitability of AE in comparison to engineered features in detecting unseen falls and 2) tightening of reconstruction based threshold to improve performance.

2. Brief Introduction to Autoencoders

An AE is an unsupervised multi-layer neural network that learns compact representation of the input data (Scholz and Vigário, 2002). An AE tries to learn an identity function such that its outputs are similar to its inputs. However, by putting constraint on the network, such as limiting the number of hidden neurons, it can discover compact representations of the data that can be used as features for other supervised or unsupervised learning tasks. An AE is often trained by using the backpropagation algorithm and consists of an encoder and decoder part. If there is one hidden layer, an AE takes the input $\mathbf{x} \in \mathbb{R}^d$ and maps it onto $\mathbf{h} \in \mathbb{R}^p$, s.t.

$$\mathbf{h} = f(\mathbf{W}\mathbf{x} + \mathbf{b}) \quad (1)$$

where \mathbf{W} is a weight matrix and \mathbf{b} is a bias term and $f(\cdot)$ is a mapping function. This step is referred to as encoding or learning latent representation, after which \mathbf{h} is mapped back to reconstruct \mathbf{y} of the same shape as \mathbf{x} , i.e.

$$\mathbf{y} = g(\mathbf{W}'\mathbf{h} + \mathbf{b}') \quad (2)$$

This step is referred to as decoding or reconstructing the input back from latent representation. An AE can be used to minimize the squared reconstruction error, L i.e.,

$$L(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|^2 \quad (3)$$

AE can learn compact and useful features if $p < d$; however, it can still discover interesting structures if $p > d$. This can be achieved by imposing a sparsity constraint on the hidden units, s.t. neurons are inactive most of the time or the average activation of each hidden neuron is close to zero. To achieve sparsity, an additional sparsity parameter is added to the objective function. Multiple layers of AEs can be stacked on top of each other to learn hierarchical features from the raw data. They are called Stacked AE (SAE). During encoding, the output of first hidden layer serves as the input to the second layer, which will learn second level hierarchical features and so on. For decoding, first the output of last layer is reconstructed at the second last hidden layer, until the original input is reconstructed.

3. Related Work

AEs can be used both in supervised and unsupervised mode for identifying falls. In a supervised classification setting, AE is used to learn representative features from both the normal and fall activities. This step can be followed by a standard machine learning classifier trained on these compressed features (Li et al., 2014b) or by a deep network (Jokanovic et al.,

2016). In the unsupervised mode or One-Class Classification (OCC) (Khan and Madden, 2014) setting, only data for normal activities is present during training. In these situations, an AE is used to learn representative features from the raw sensor data of normal activities. This step is followed by either employing (i) a discriminative model by using one-class classifiers or (ii) a generative model with appropriate threshold based on reconstruction error, to detect falls and normal activities.

Much work has been done in evaluating the feasibility of learning generic representations through AEs for general activity recognition and fall detection tasks. Plötz et al. (Plötz et al., 2011) explore the potential of discovering universal features for context-aware application using wearable sensors. They present several feature learning approaches using PCA and AE and show their superior performance in comparison to standard features across a range of activity recognition applications. Budiman et al. (Budiman et al., 2014) use SAEs and marginalized SAE to infer generic features in conjunction with neural networks and Extreme Learning Machines (ELM) as the supervised classifiers to perform pose-based action recognition. Li et al. (Li et al., 2014a) compare SAE, Denoising AE and PCA for unsupervised feature learning in activity recognition using smartphone sensors. They show that traditional features perform worse than the generic features inferred through autoencoders. Jokanovic et al. (Jokanovic et al., 2016) use SAE to learn generic lower dimensional features and use softmax regression classifier to identify falls using radar signals. Other researchers (Jankowski et al., 2015; Wang, 2016) have used AEs to reduce the dimensionality of domain specific features prior to applying traditional supervised classification models or deep belief networks.

AEs have also been extensively used in anomaly detection. Japkowicz et al. (Japkowicz et al., 1995) present the use of AE for novelty detection. For noiseless data, they propose to use a reduced percentage of maximum of reconstruction error as a threshold to identify outliers. For noisy data, they propose to identify both the intermediate positive and negative regions and subsequently optimizing the threshold until a desired accuracy is achieved. Manevitz and Yousef (Manevitz and Yousef, 2007) present an AE approach to filter documents and report better performance than traditional classifiers. They report to carry out certain type of uniform transformation before training the network to improve the performance. They discuss that choosing the appropriate threshold to identify normal documents is challenging and present several variants. The method that worked the best is to tighten the threshold sufficiently to disallow the classification of the highest 25 percentile error cases from the training set. Erfani et al. (Erfani et al., 2016) present a hybrid approach to combine the AEs and one-class SVM (*OSVM*) for anomaly detection in high-dimensional and large-scale applications. They first extract generic features using SAE and train an *OSVM* with linear kernel on learned features from SAE. They also use SAE as a one-class classifier by setting the threshold to be 3 times of standard deviation away from the mean. Their results show comparable results in comparison to AE based anomaly classifier but the training and testing time greatly reduced. Sakurda and Yairi (Sakurada and

Yairi, 2014) show the use of AE in anomaly detection task and compare it with PCA and Kernel PCA. They demonstrate that the AE can detect subtle anomalies that PCA could not and is less complex than Kernel PCA.

Ensembles of AE have been used to learn diverse feature representations, mainly in the supervised settings. Ithapu et al. (Ithapu et al., 2014) present an ensemble of SAE by presenting it with randomized inputs and randomized sample sets of hyper-parameters from a given hyperparameter space. They show that their approach is more accurately related to different stages of Alzheimer’s disease and leads to efficient clinical trials with very less sample estimates. Reeve and Gavin (Reeve and Brown, 2015) present a modular AE approach that consists of M AE modules trained separately on different data representations and the combined result is defined by taking an average of all the modules present. Their results on several benchmark datasets show improved performance in comparison to baseline of bootstrap version of the AE. Dong and Japkowicz (Dong and Japkowicz, 2016) present a supervised and unsupervised ensemble approach for stream learning that uses multi-layer neural networks and AE. They train their models from multi-threads which evolve with data streams, the ensemble of the AE is trained using only the data from positive class and is accurate when anomalous training data are rare. Their method performs better as compared to the state-of-the-art in terms of detection accuracy and training time for the datasets.

The research on using AE for supervised case show that AE can successfully learn generic features from raw sensor data for activity recognition. We observe that AE can be effectively used for anomaly detection tasks and their ensembles can perform better than a single AE. In this paper, fall detection problem is formulated as an OCC or anomaly detection, where abundant data for normal activities is available during training and none for falls. We investigate the utility of features learned through AE and their ensembles for the task of fall detection.

4. Autoencoder Ensemble for Detecting Unseen Falls

In the absence of training data for falls, a fall can be detected by training an AE/SAE on only the normal activities to learn representative features from a wearable device. These features can be fed to standard OCC algorithms to detect a test sequence as a normal activity or not (a fall in our case). Alternatively, based on the training data, a threshold can be set on the reconstruction error of the AE/SAE to identify a test sequence as an abnormal activity (a fall in our case) if its reconstruction error is high. Intuitively, this would mean that the test sequence is very different from the training data comprising of normal activities.

4.1. Monolithic Autoencoders

Figure 1 shows the AE/SAE for training normal activities using raw sensor data from a three-channel accelerometer and gyroscope. The raw sensor readings coming from each of the channels of accelerometer (a_x, a_y, a_z) and gyroscope ($\omega_x, \omega_y, \omega_z$) are combined and presented as input to the AE/SAE. For a sliding window of a fixed length (n samples), $\mathbf{a}_x = [a_x^1, a_x^2, \dots, a_x^n]$, $\mathbf{a}_y = [a_y^1, a_y^2, \dots, a_y^n]$, and so on. The feature

vector for a time window is constructed by concatenating these sensor readings as $\mathbf{f} = [\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z, \omega_x, \omega_y, \omega_z]^T$. We call this feature learning approach as *monolithic* because it combines raw sensor data from different channels as one input to an AE.

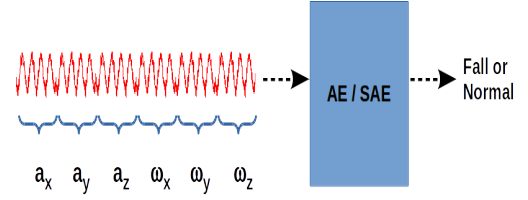
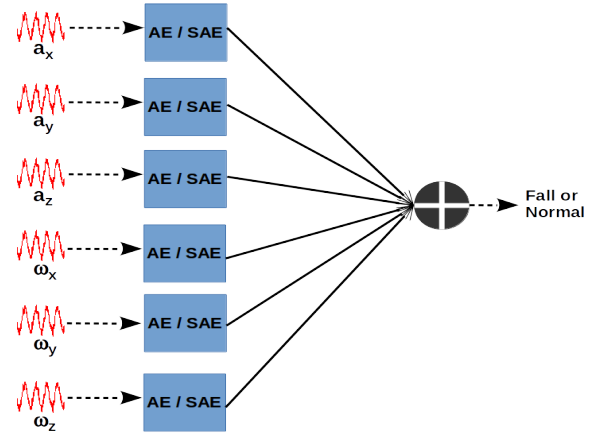
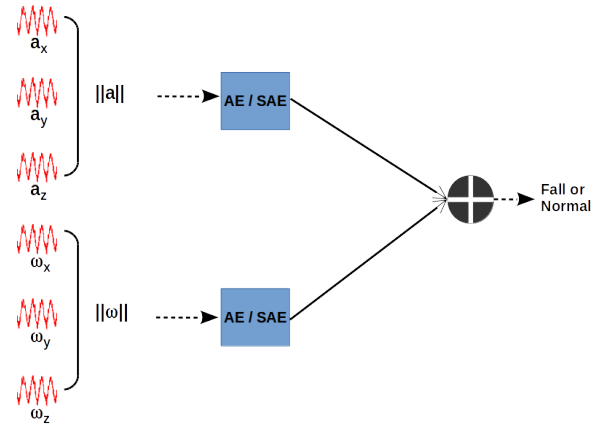


Fig. 1: Monolithic AE for detecting unseen falls.



(a) Ensemble of 6 separate channels of accelerometer and gyroscope.



(b) Ensemble of 2 channels of the magnitude of accelerometer and gyroscope.

Fig. 2: Channel-wise AE for detecting unseen falls.

4.2. Channel-wise Autoencoders

Li et al. (Li et al., 2014a) present the use of ensemble of SAE by extracting generic features per each of the three accelerometer channels and additional channel for the magnitude of the accelerometer vector in 3-dimensions. They extract fixed number of features for each of these 4 channels and concatenate

them. Supervised classification methods are then used on these extracted features. This setting can work for supervised classification but not in OCC scenario. In our case, separate AEs can be trained on data coming from different channels of the sensors; wherein each AE can detect an unseen fall based on reconstruction error and their overall result can be combined to take a final decision. We propose to use two types of channel-wise ensemble strategies for detecting unseen falls as follows:

- **Six Channel Ensemble (6 – CE):** For each of the 6 channels of an accelerometer and a gyroscope (i.e., $\mathbf{a}_x, \mathbf{a}_y, \mathbf{a}_z, \omega_x, \omega_y, \omega_z$), 6 separate AE/SAE are trained to learn a compact representation for each channel. A decision threshold can be employed on each of these 6 AEs to decide whether a test sample is normal or a fall.
- **Two Channel Ensemble (2 – CE):** Alternatively, we can compute the magnitude of the 3 accelerometer channels and that of the 3 gyroscope channels. The magnitude vector gives direction invariant information. We train two separate AE/SAE to learn a compact representation for each of the two magnitude channels. Thresholding the reconstruction error on these two channels can be used to decide whether a test sample is normal activity or a fall.

For a given test sample, the 6 – CE will give 6 different decisions and the 2 – CE give 2 decisions. These decisions can be combined by majority voting to arrive at a final decision; as a convention, ties are considered as falls. For simplicity, we keep the hyper-parameters for each AE/SAE corresponding to a channel as the same. The ensemble approach can be faster than the monolithic approach because AE/SAE per channel utilizes less amount of data in comparison to the combined 6 channel data to a single AE/SAE. Figure 2 shows the graphical representation of the 6 – CE and 2 – CE approaches.

5. Optimizing the Threshold on the Reconstruction Error

Our assumption is that fall data is rare and is not present during training phase. Therefore, we train AE and SAE and their ensembles on raw sensor data to learn a compact representation of normal activities. The typical approach to identify a fall as an anomaly is to set the threshold on the reconstruction error to the maximum of the reconstruction error on the full training data. We call this threshold *MaxRE*. During testing, any sample that has a reconstruction error greater than this value can be identified as a fall. However, sensor readings are not perfect and may contain spurious data, which can affect this threshold (Khan et al., 2014). Due to the presence of outliers present in the training data, *MaxRE* is often too high, which could result in many of the falls being missed in test time. Another approach is to tighten the reconstruction error, setting it, for instance, as 3 standard deviations away from the mean of the training data reconstruction error (Erfani et al., 2016). We name this threshold as *StdRE*. Below, we present two new approaches to tighten the threshold such that it can better identify unseen falls. These methods are similar to finding an optimal operating point on an ROC curve by reducing false negatives at the cost of false

alarms. However, in a OCC framework, it is difficult to adopt a traditional ROC approach because of the unavailability of the validation data for the negative class. The proposed methods overcome these difficulty by removing outliers from the training data prior to setting a threshold based on the training data.

5.1. Reduced Reconstruction Error

The raw sensor data may not be perfect and may contain spurious or incorrectly labeled readings (Khan et al., 2014). If an AE/SAE is trained on normal activities on such data, the reconstruction error for some of the samples of the training set may be very high. In this case, choosing the maximum of reconstruction error as the threshold to identify falls may lead to accepting most of the falls as normal activities. Manevitz and Yousef (Manevitz and Yousef, 2007) suggest to tighten this threshold; however they choose the threshold by an amount heuristically related to the percentage of near zero vectors in the training set. In our case, we did not get zero vector reconstruction error; therefore, such strategy cannot be directly applied. Khan et al. (Khan et al., 2014) propose to use the concept of quartiles from descriptive statistic to adjust the threshold in the context of hidden Markov models. Here we apply their idea to tighten the threshold on the reconstruction error. Given the reconstruction error on the training data comprising of only instances of normal activities, the lower quartile (Q_1), the upper quartile (Q_3) and the inter-quartile range ($IQR = Q_3 - Q_1$), a point P is qualified as an outlier of the normal class, if

$$P > Q_3 + \Omega \times IQR \quad \parallel \quad P < Q_1 - \Omega \times IQR \quad (4)$$

where Ω is the rejection rate that represents the percentage of data points that are within the non-extreme limits. Based on Ω , the extreme values of reconstruction error that represents spurious training data can be removed and a threshold can be chosen as the maximum of the remaining reconstruction errors. We call this method as Reduced Reconstruction Error (*RRE*). The value of Ω can be found experimentally or set to remove a small fraction of the normal activities data.

5.2. Inliers Reconstruction Error

We propose another method to adjust the threshold based on reconstruction error. In this method, we first train an AE/SAE on full normal data and then remove the corresponding anomalous training instances based on Ω from the training set (as discussed in the previous section). After this step, we are left with training data sans the outlier instances. Then, we train a new AE on this reduced data comprising of just the inliers. The idea is that the variance of reconstruction error for such inlier data will not be too high and its maximum can serve as the new threshold. We call this method as Inliers Reconstruction Error (*IRE*). For the Channel-wise ensemble approach, each AE/SAE is trained only using the raw sensor data from a specific channel of normal activities, then various thresholds, i.e., *MaxRE*, *StdRE*, *RRE* and *IRE* are computed for each channel separately. During testing, for a given threshold method, the final decision is taken as the majority voting outcome of all the AE/SAE.

6. Experimental Analysis

We show our results on two activity recognition datasets that includes fall events collected via wearable devices.

6.1. Datasets

6.1.1. German Aerospace Center (DLR) (Nadales, 2010)

This dataset is collected using an Inertial Measurement Unit with a sampling frequency of 100 Hz. The dataset contains samples from 19 people of both genders of different age groups. The data is recorded in indoor and outdoor environments under semi-natural conditions. The sensor is placed on the belt either on the right or the left side of the body or in the right pocket in different orientations. The dataset contains labelled data of the following 7 activities: Standing, Sitting, Lying, Walking (up/downstairs, horizontal), Running/Jogging, Jumping and Falling. One of the subjects did not perform fall activity; therefore, their data is omitted from the analysis.

6.1.2. Coventry Dataset (COV) (Ojetola et al., 2015)

This dataset is collected using two SHIMMERTM sensor nodes strapped to the chest and thighs of subjects with a sampling frequency of 100 Hz. Two protocols were followed to collect data from subjects. In Protocol 1, data for six types of fall scenarios are captured (forward, backward, right, left, real fall-backward and real fall forward) and a set of ADL (standing, lying, sitting on a chair or bed, walking, crouching, near falls and lying). Protocol 2 involved ascending and descending stairs. 42 young healthy individuals simulated various ADL and fall scenarios (32 in Protocol 1 and 10 in Protocol 2). These data from different types of falls are joined together to make one separate class for falls. The subjects for Protocol 2 did not record corresponding fall data; therefore, that data is not used. In our analysis, we used accelerometer and gyroscope data from the sensor node strapped to the chest.

6.2. Experimental Setup

For both the datasets, the raw sensor data is processed using a 50% overlapping sliding window. The time window size was 1.28 seconds for the DLR dataset and 2.56 seconds for the COV dataset to compare with the work of Khan (Khan, 2016). We tested two types of AE for the analysis; one with a single hidden layer and other with three layered SAE. For the monolithic AE, the raw data within a time window for each of the sensor is concatenated, which leads to 768 input layer neurons for the DLR dataset and 1536 input layer neurons for the COV dataset. The number of hidden neurons (i.e., the number of generic features learned) is set to 31 to compare them with the work of Khan et al. (Khan, 2016). For the monolithic SAE, the number of hidden neurons in the first layer is chosen to be half of the number of input neurons, i.e. 384 for DLR dataset and 768 for COV dataset and the second layer has 31 number of features. For the channel-wise ensemble method, each channels is fed to the AE/SAE separately. Therefore, the number of neurons in the input layer per AE/SAE is set to 128 for DLR dataset and 256 for COV dataset. Rest of the configuration is kept same as for the monolithic features case. The number of training epochs

is fixed to 50 for all the different autoencoders during the experiment. Rest of the parameters such as sparsity parameters, activation function etc., are kept at the default values (MATLAB, 2016). Compressed features learned through monolithic AE and SAE are further used to train *OSVM* and One-class nearest neighbour (*OCNN*) classifiers (Khan and Madden, 2014) to identify unseen falls.

6.2.1. Internal Cross-Validation

For *OCNN*, the number of nearest neighbours to identify an outlier is kept as 1. *OSVM* has a parameter ν to control the trade-off between ensuring most training examples are in the positive class and minimizing the weights in the score function. The value of this hyper parameter was tuned as follows.

An internal 3-fold cross validation was employed to optimize the parameters ν for the *OSVM* and Ω for the *RRE* and *IRE* thresholding methods. In the absence of validation data for falls, it is not straight forward how these hyper parameters could be optimized. We use the IQR technique (discussed in Section 5.1) to remove a small amount of outlier data from the normal class. These outliers are not actual falls but very different from the normal activities and may act as proxies for falls and in turn will help in optimizing the parameters. This parameter is set once (to $\Omega = 3$) before the training commences. As it happens in training, during the testing phase, we expect a skewed distribution of falls. Therefore standard performance metrics such as accuracy may not be appropriate. To deal such as a case, we use the geometric mean (*gmean*) (Khan et al., 2014) as the performance metric to report results and optimize the parameters. *gmean* is defined as the square root of the multiplication of true positive and true negative rate. the value of *gmean* varies from 0 to 1, where the means a perfect classification among falls and normal activities.

6.3. Results

Along with 6 channel raw data to train different classifiers to detect unseen falls, we also use 2 channel magnitude data to train these classifiers. Therefore, in the experiment we compare the following different classifiers:

- Two types of AE, i.e. single layer AE and three layered SAE.
- Four types of thresholding methods i.e. *MaxRE*, *StdRE*, *RRE* and *IRE*.
- Two types of feature learning techniques - (i) monolithic and (ii) two types of channel-wise ensemble.
- *OCNN* and *OSVM* trained on features learned from AE and SAE.
- *OCNN* and *OSVM* trained on engineered features from Khan (Khan, 2016).

This comparison gives 48 different classifiers trained using the raw sensor data and 2 classifiers trained using engineered features. Tables 1 and 2 show the *gmean* values for the DLR datasets when 6-channels and 2-channels data is used to learn features from different AES and various thresholding

techniques and other classification methods are employed to identify unseen falls. We observe that the single layered AE with channel-wise ensemble and *RRE* thresholding gives the best value for *gmean* for both the 6-*CE* and 2-*CE* approaches. The best performance metric for 2-*CE* is slightly better than the 6-*CE*. We observe similar results for the COV datasets (see Tables 5 and 6). For the 6-*CE* approach, ensemble of channel-wise feature learning with single layer AE and *RRE* threshold performs the best. However, for the 2-*CE* approach for COV data, all the monolithic and channel-wise ensemble with single layer AE and three-layered SAE with *RRE* threshold and monolithic single layer AE with *IRE* threshold performed equivalent.

Table 3 shows *gmean* values on the DLR dataset when features are learned from the raw data from 6 and 2 channels and *OCNN* and *OSVM* are trained to identify unseen falls. Table 4 shows the results on extracting engineered features and training *OCNN* and *OSVM* on them. We observe that performance of both the one-class classifiers that are trained on learned features from both AE and SAE and engineered features is worse than the single layered AE with *RRE* threshold on both 6 and 2 channel raw data. Similar results are shown in Tables 7 and 4 for COV datasets with the only exception that features learned from 2 channel magnitude data using single layer AE when trained with *OSVM* give comparable results to the best methods discussed above for COV dataset. The *OCNN* approach did not perform well for both the datasets.

Table 1: *gmean* values for DLR dataset (6 channels)

Features Types	AE Type	Thresholding			
		<i>MaxRE</i>	<i>StdRE</i>	<i>RRE</i>	<i>IRE</i>
Monolithic	AE	0	0.278	0.790	0.648
	SAE	0	0.278	0.746	0.286
Channel-wise	AE	0	0.463	0.846	0.719
	SAE	0	0.490	0.753	0.669

Table 2: *gmean* values for DLR dataset (2 channels)

Features Type	AE Type	Thresholding			
		<i>MaxRE</i>	<i>StdRE</i>	<i>RRE</i>	<i>IRE</i>
Monolithic	AE	0	0.251	0.776	0.148
	SAE	0	0	0.699	0.586
Channel-wise	AE	0.036	0.624	0.867	0.502
	SAE	0	0.524	0.764	0.749

Table 3: *gmean* values for DLR dataset on training one-class classifiers on features learned from AE and SAE on different channels of raw data.

Channel Type	Feature Type	<i>OCNN</i>	<i>OSVM</i>
6- <i>CE</i>	AE	0.562	0.071
	SAE	0.476	0.479
2- <i>CE</i>	AE	0.593	0.284
	SAE	0.402	0

We also compare our results with two supervised classifiers. We use all the raw training data for falls, learn (31) generic

Table 4: *gmean* values on DLR and COV datasets when one-class classifiers are trained on engineered features

Datasets	<i>OCNN</i>	<i>OSVM</i>
DLR	0.381	0.163
COV	0.490	0.084

features using AE and feed them to SVM and Random Forest (RF) classifiers. The number of trees are kept as 200 in the RF (Stone and Skubic, 2015). The soft margin hyper-parameter for SVM is tuned using an internal cross-validation that uses all the labelled data for normal activities and falls. We use two variants of each of the dataset; in the first one, raw data from all the 6 channels is combined for a given sample and fed to AE (DLR-6 and COV-6). In the second, for a given sample, we compute the magnitudes of acceleration and gyroscope signals and feed them to the AE to learn features (DLR-2 and COV-2). The results in terms of *gmean* are show in table 8. We observe that the performance of supervised classifiers are very poor in comparison to the OCC approach proposed in this paper. We also extract 31 engineered features (Khan, 2016) from the raw sensor data and show the results (DLR-Eng and COV-Eng). In the supervised case, using the learned features, most of the time a fall is wrongly classified as a normal activity. Whereas, using engineered features, we get high detection rates. This shows that for supervised classification, AE does not learn discriminative features in comparison to the engineered features.

Table 5: *gmean* values for COV dataset (6 channels)

Features Type	AE Type	Thresholding			
		<i>MaxRE</i>	<i>StdRE</i>	<i>RRE</i>	<i>IRE</i>
Monolithic	AE	0.019	0.747	0.770	0.774
	SAE	0.015	0.740	0.773	0.772
Channel-Wise	AE	0.368	0.621	0.796	0.745
	SAE	0	0.240	0.584	0.536

Table 6: *gmean* values for COV dataset (2 channels)

Features Type	AE Type	Thresholding			
		<i>MaxRE</i>	<i>StdRE</i>	<i>RRE</i>	<i>IRE</i>
Monolithic	AE	0.035	0.762	0.790	0.801
	SAE	0.014	0.753	0.793	0.667
Channel-Wise	AE	0.311	0.762	0.793	0.786
	SAE	0.322	0.763	0.800	0.779

7. Conclusions and Future Work

A fall is a rare event; therefore, it is difficult to build classification models using supervised algorithms. Another problem is to extract discriminative features in the absence of fall data for training the right classifiers. In this paper, we presented solutions to deal with these issues. Firstly, we formulated a fall detection problem as a one-class classification or outlier detection

Table 7: *gmean* values for COV dataset on training one-class classifiers on features learned from AE and SAE on different channels of raw data.

Channel Type	Feature Type	OCNN	OSVM
6 – CE	AE	0.481	0.271
	SAE	0.539	0.215
2 – CE	AE	0.594	0.807
	SAE	0.510	0.047

Table 8: *gmean* value for supervised classification using learned and engineered features

Datasets	SVM	RF
DLR-6	0.127	0.097
DLR-2	0.023	0.023
DLR-Eng	0.929	0.622
COV-6	0.139	0.212
COV-2	0.538	0.623
COV-Eng	0.884	0.848

problem. Secondly, we presented the use of AE, more specifically a novel way to train separate AE for each channel of the wearable sensor to learn generic features. We proposed threshold tightening methods to identify unseen falls accurately. This work gives us useful insights that an ensemble of generic features learned from different channels using AE along with *RRE* optimized threshold is a useful technique to identify falls in a scenario when the falls are not present during training. In future work, we are exploring extreme value theory and combining it with the proposed approaches to identify unseen falls.

Acknowledgments

This work was partially supported by the AGE-WELL NCE Trainee Award Program and by the Canadian Consortium on Neurodegeneration in Aging (CCNA).

References

- Budiman, A., Fanany, M.I., Basaruddin, C., 2014. Stacked denoising autoencoder for feature representation learning in pose-based action recognition, in: 2014 IEEE 3rd Global Conference on Consumer Electronics (GCCE), pp. 684–688.
- Dong, Y., Japkowicz, N., 2016. Advances in Artificial Intelligence: 29th Canadian Conference on Artificial Intelligence, Canadian AI 2016, Victoria, BC, Canada, May 31 - June 3, 2016. Proceedings. Springer International Publishing, Cham. chapter Threaded Ensembles of Supervised and Unsupervised Neural Networks for Stream Learning. pp. 304–315.
- El-Bendary, N., Tan, Q., Pivot, F.C., Lam, A., 2013. Fall detection and prevention for the elderly: A review of trends and challenges. International Journal on Smart Sensing and Intelligent Systems 6, 1230–1266.
- Erfani, S.M., Rajasegarar, S., Karunasekera, S., Leckie, C., 2016. High-dimensional and large-scale anomaly detection using a linear one-class {SVM} with deep learning. Pattern Recognition 58, 121 – 134.
- Igual, R., Medrano, C., Plaza, I., 2013. Challenges, issues and trends in fall detection systems. BioMedical Engineering OnLine 12, 1–24.
- Ithapu, V.K., Singh, V., Okonkwo, O., Johnson, S.C., 2014. Randomized denoising autoencoders for smaller and efficient imaging based ad clinical trials, in: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2014. Springer, pp. 470–478.
- Jankowski, S., Szymanski, Z., Dziomin, U., Mazurek, P., Wagner, J., 2015. Deep learning classifier for fall detection based on ir distance sensor data, in: Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 2015 IEEE 8th International Conference on, IEEE. pp. 723–727.
- Japkowicz, N., Myers, C., Gluck, M., 1995. A novelty detection approach to classification, in: Proceedings of the 14th international joint conference on Artificial intelligence-Volume 1, Morgan Kaufmann Publishers Inc.. pp. 518–523.
- Jokanovic, B., Amin, M., Ahmad, F., 2016. Radar fall motion detection using deep learning, in: Proc. IEEE Radar Conference.
- Khan, S., 2016. Classification and decision-theoretic framework for detecting and reporting unseen falls. Ph.D. thesis. University of Waterloo.
- Khan, S.S., Karg, M.E., Kulić, D., Hoey, J., 2014. X-factor HMMs for detecting falls in the absence of fall-specific training data, in: et al., L.P. (Ed.), Proceedings of the 6th International Work-conference on Ambient Assisted Living (IWAAL 2014), Springer International Publishing Switzerland, Belfast, U.K.. pp. 1–9.
- Khan, S.S., Madden, M.G., 2014. One-class classification: taxonomy of study and review of techniques. The Knowledge Engineering Review 29, 345–374.
- Li, Y., Shi, D., Ding, B., Liu, D., 2014a. Mining Intelligence and Knowledge Exploration: Second International Conference, MIKE 2014, Cork, Ireland, December 10–12, 2014. Proceedings. Springer International Publishing, Cham. chapter Unsupervised Feature Learning for Human Activity Recognition Using Smartphone Sensors. pp. 99–107.
- Li, Y., Shi, D., Ding, B., Liu, D., 2014b. Unsupervised feature learning for human activity recognition using smartphone sensors, in: Mining Intelligence and Knowledge Exploration. Springer, pp. 99–107.
- Manevitz, L., Yousef, M., 2007. One-class document classification via neural networks. Neurocomputing 70, 1466 – 1481. Advances in Computational Intelligence and Learning 14th European Symposium on Artificial Neural Networks 2006 14th European Symposium on Artificial Neural Networks 2006.
- MATLAB, 2016. Train autoencoder. <http://www.mathworks.com/help/nnet/ref/trainautoencoder.html>. Accessed on 23rd September, 2016.
- Nadales, M.J.V., 2010. Recognition of Human Motion Related Activities from Sensors. Master’s thesis. University of Malaga and German Aerospace Cener.
- Ojetola, O., Gaura, E., Brusey, J., 2015. Data set for fall events and daily activities from inertial sensors, in: Proceedings of the 6th ACM Multimedia Systems Conference, ACM, New York, NY, USA. pp. 243–248.
- Organization, W.H., 2016. Falls fact sheet, reviewed september 2016. <http://www.who.int/mediacentre/factsheets/fs344/en/>. Accessed on 11th October 2016.
- Pannurat, N., Thiemjarus, S., Nantajeewarawat, E., 2014. Automatic fall monitoring: a review. Sensors 14, 12900–12936.
- Plötz, T., Hammerla, N.Y., Olivier, P., 2011. Feature learning for activity recognition in ubiquitous computing, in: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Two, AAAI Press. pp. 1729–1734.
- Ravi, N., Dandekar, N., Mysore, P., Littman, M.L., 2005. Activity recognition from accelerometer data, in: Proceedings of the 17th conference on Innovative applications of artificial intelligence - Volume 3, AAAI Press. pp. 1541–1546.
- Reeve, H.W.J., Brown, G., 2015. Modular autoencoders for ensemble feature extraction, in: NIPS 2015 Workshop on Feature Extraction: Modern Questions and Challenges.
- Sakurada, M., Yairi, T., 2014. Anomaly detection using autoencoders with nonlinear dimensionality reduction, in: Proceedings of the MLSDA 2014 2Nd Workshop on Machine Learning for Sensory Data Analysis, ACM, New York, NY, USA. pp. 4:4–4:11.
- Scholz, M., Vigário, R., 2002. Nonlinear pca: a new hierarchical approach., in: ESANN, pp. 439–444.
- Stone, E., Skubic, M., 2015. Fall detection in homes of older adults using the microsoft kinect. Biomedical and Health Informatics, IEEE Journal of 19, 290–301.
- Wang, L., 2016. Recognition of human activities using continuous autoencoders with wearable sensors. Sensors 16, 189.